Searching within **The ACM Digital Library** with **Advanced Search**: (Abstract:compiling and Abstract:code) (start a new search)

Found **116** of **258,874**

REFINE YOUR SEARCH

Refine by Keywords

SEARCH

Discovered Terms

Refine by People

Names

Institutions

Authors

Reviewers

Refine by Publications

Publication Year

Publication Names

ACM Publications

All Publications

Content Formats

Publishers

Refine by Conferences

Sponsors

Events

Proceeding Series

ADVANCED SEARCH

Advanced Search

FEEDBACK

Please provide us with feedback

Found **116** of **258,874**

Search Results • Related Journals • Related Magazines • Related SIGs • Related Conferences

Results 1 - 20 of 116

Sort by relevance in expanded form

Save results to a Binder

Result page: **1** 2 3 4 5 6 next >>

1 Offset assignment using simultaneous variable coalescing

Desiree Ottoni, Guilherme Ottoni, Guido Araujo, Rainer Leupers

November 2006 **Transactions on Embedded Computing Systems (TECS)** , Volume 5 Issue 4

Publisher: ACM Request Permissions

Full text available: Pdf (554.99 KB) Additional Information: full citation, abstract, references, index terms

Bibliometrics: Downloads (6 Weeks): 17, Downloads (12 Months): 55, Downloads (Overall): 217, Citation Count: 0

The generation of efficient addressing code is a central problem in compiling for processors with restricted addressing modes, like digital signal processors (DSPs). Offset assignment (OA) is the problem of allocating scalar variables to memory, so as ...

Keywords: DSPs, Stack offset assignment, address registers, autoincrement addressing modes, register allocation, variable coalescing

2 The design of an integrated support software system

Arra Avakian, Sam Haradhvala, Bruce Knobe

June 1982 **SIGPLAN '82:** Proceedings of the 1982 SIGPLAN symposium on Compiler construction

Publisher: ACM

Full text available: Pdf (850.53 KB) Additional Information: full citation, abstract, references, cited by, index terms

Bibliometrics: Downloads (6 Weeks): 6, Downloads (12 Months): 13, Downloads (Overall): 115, Citation Count: 2

This paper describes some of the interesting features of a large integrated support software system. The system was built to support the development, on an IBM 370, of an extremely large Pascal program to be run on a network of Intel 8086 microprocessors. ...

Also published in:

June 1982 **SIGPLAN Notices** Volume 17 Issue 6

3 Compiling programs for a linear systolic array

Ping-Sheng Tseng

June 1990 **PLDI '90:** Proceedings of the ACM SIGPLAN 1990 conference on Programming language design and implementation

Publisher: ACM Request Permissions

Full text available: Pdf (1.07 MB) Additional Information: full citation, abstract, references, cited by, index terms

Bibliometrics: Downloads (6 Weeks): 8, Downloads (12 Months): 24, Downloads (Overall): 257, Citation Count: 4

This paper describes an AL compiler for the Warp systolic array. AL is a programming language in which the user programs a systolic array as if it were a sequential computer and relies on the compiler to generate parallel code. This paper introduces ...

Also published in:

June 1990 **SIGPLAN Notices** Volume 25 Issue 6

4 An Esterel processor with full preemption support and its worst case reaction time analysis

Xin Li, Jan Lukoschus, Marian Boldt, Michael Harder, Reinhard von Hanxleden

September 2005 **CASES '05:** Proceedings of the 2005 international conference on Compilers, architectures and synthesis for embedded systems

Publisher: ACM

Full text available: Pdf (630.56 KB) Additional Information: full citation, abstract, references, cited by, index terms

Bibliometrics: Downloads (6 Weeks): 8, Downloads (12 Months): 32, Downloads (Overall): 227, Citation Count: 6

The concurrent synchronous language Esterel allows programmers to treat reactive systems in an abstract, concise manner. An Esterel program is typically first translated into other, non-synchronous high-level languages, such as VHDL or C, and then compiled ...

Keywords: Esterel, WCET, reaction time analysis, reactive processing, synchronous languages

5 Representing Java classes in a typed intermediate language

Christopher League, Zhong Shao, Valery Trifonov

September 1999 **ICFP '99:** Proceedings of the fourth ACM SIGPLAN international conference on Functional programming

Publisher: ACM Request Permissions

Full text available: Pdf (1.81 MB) Additional Information: full citation, abstract, references, cited by, index terms

Bibliometrics: Downloads (6 Weeks): 7, Downloads (12 Months): 27, Downloads (Overall): 256, Citation Count: 12

We propose a conservative extension of the polymorphic lambda calculus (*F*ω) as an intermediate language for compiling languages with name-based class and interface hierarchies. Our extension enriches standard *F*ω ...

Also published in:

September 1999 **SIGPLAN Notices** Volume 34 Issue 9

6 Optimization of the compiling of subscripted array references

Irving B. Elliott

November 1978 **SIGPLAN Notices** , Volume 13 Issue 11

Publisher: ACM

Full text available: Pdf (348.90 KB) Additional Information: full citation, abstract

Bibliometrics: Downloads (6 Weeks): 5, Downloads (12 Months): 8, Downloads (Overall): 21, Citation Count: 0

A method is described for compiling higher-order-language references to array elements into optimum in-line executable code. Optimization is achieved by resolving parts of the required subscripting calculation, e.g., those which involve constant values, ...

7 An improved mixture rule for pattern matching

J. Ophel

June 1989 **SIGPLAN Notices** , Volume 24 Issue 6

Publisher: ACM

Full text available: Pdf (280.26 KB) Additional Information: full citation, abstract, index terms

Bibliometrics: Downloads (6 Weeks): 3, Downloads (12 Months): 8, Downloads (Overall): 93, Citation Count: 0

An improved mixture rule for compiling the pattern matching mechanism used in function definitions for ML, Miranda and Hope is presented. The new rule produces better code for patterns with mixed constructors and variables in multiple columns.

8 Fast Paths in Concurrent Programs

Wen Xu, Sanjeev Kumar, Kai Li

September 2004 **PACT '04:** Proceedings of the 13th International Conference on Parallel Architectures and Compilation Techniques

Publisher: IEEE Computer Society

Full text available: Pdf (322.08 KB) Additional Information: full citation, abstract, references

Bibliometrics: Downloads (6 Weeks): 6, Downloads (12 Months): 14, Downloads (Overall): 80, Citation Count: 0

Compiling concurrent programs to run on a sequential processor presents a difficult tradeoff between execution time and size of generated code. On one hand, the process-based approach to compilation generates reasonable sized code but incurs significant ...

9 Generating sequential code from parallel code

J. Ferrante, M. Mace, B. Simons

June 1988 **ICS '88:** Proceedings of the 2nd international conference on Supercomputing

Publisher: ACM Request Permissions

Full text available: Pdf (1.12 MB) Additional Information: full citation, abstract, references, cited by, index terms

Bibliometrics: Downloads (6 Weeks): 1, Downloads (12 Months): 17, Downloads (Overall): 179, Citation Count: 8

We consider the problem of generating sequential code for parallel programs written in a language which contains a FORALL operator, predicates and statements. This problem can arise when compiling for a multiprocessor where each processor is sequential, ...

10 A methodology for analyzing the temporal evolution of novice programs based on semantic components

Christopher D. Hundhausen, Jonathan L. Brown, Sean Farley, Daniel Skarpas

September 2006 **ICER '06:** Proceedings of the 2006 international workshop on Computing education research

Publisher: ACM Request Permissions

Full text available: Pdf (317.69 KB) Additional Information: full citation, abstract, references, index terms

Bibliometrics: Downloads (6 Weeks): 4, Downloads (12 Months): 35, Downloads (Overall): 187, Citation Count: 0

Empirical studies of novice programming typically rely on code solutions or test responses as the basis of their analyses. While such data can provide insight into novice programming knowledge, they say little about the programming processes in which ...

Keywords: algorithm visualization, novice programming environments, programming process, semantic components, video analysis

11 Automatic array alignment in data-parallel programs

Siddhartha Chatterjee, John R. Gilbert, Robert Schreiber, Shang-Hua Teng

March 1993 **POPL '93:** Proceedings of the 20th ACM SIGPLAN-SIGACT symposium on Principles of programming languages

Publisher: ACM Request Permissions

Full text available: Pdf (1.34 MB) Additional Information: full citation, abstract, references, cited by, index terms

Bibliometrics: Downloads (6 Weeks): 4, Downloads (12 Months): 28, Downloads (Overall): 187, Citation Count: 30

Data-parallel languages like Fortran 90 express parallelism in the form of operations on data aggregates such as arrays. Misalignment of the operands of an array operation can reduce program performance on a distributed-memory parallel machine by requiring ...

12 Name resolutions using a microprogrammed interpretive technique

Stanley Habib

September 1973 **MICRO 6:** Conference record of the 6th annual workshop on Microprogramming

Publisher: ACM

Full text available: Pdf (455.42 KB) Additional Information: full citation, abstract, references, cited by, index terms

Bibliometrics: Downloads (6 Weeks): 1, Downloads (12 Months): 2, Downloads (Overall): 60, Citation Count: 1

During the compiling process, certain name resolutions must be made to ascertain whether a particular name has been used previously. If it has been used previously, the attributes this particular name possesses must be noted for use during the execution ...

13 On programming parallel computers

Leslie Lamport

January 1975 Proceedings of the conference on Programming languages and compilers for parallel and vector machines

Publisher: ACM

Full text available: Pdf (1.05 MB) Additional Information: full citation, abstract, references, index terms

Bibliometrics: Downloads (6 Weeks): 6, Downloads (12 Months): 22, Downloads (Overall): 174, Citation Count: 0

In this paper, I will make some general observations about how computers should be programmed, and how programs should be compiled. I will restrict my attention to programming computers to solve numerical analysis problems, although most of my remarks ...

Also published in:

March 1975 **SIGPLAN Notices** Volume 10 Issue 3

14 Parsing and compiling using Prolog

Jacques Cohen, Timothy J. Hickey

March 1987 **Transactions on Programming Languages and Systems (TOPLAS)** , Volume 9 Issue 2

Publisher: ACM Request Permissions

Full text available: Pdf (2.83 MB) Additional Information: full citation, abstract, references, cited by, index terms, review

Bibliometrics: Downloads (6 Weeks): 20, Downloads (12 Months): 162, Downloads (Overall): 910, Citation Count: 6

This paper presents the material needed for exposing the reader to the advantages of using Prolog as a language for describing succinctly most of the algorithms needed in prototyping and implementing compilers or producing tools that facilitate this ...

15 The Opie compiler from row-major source to Morton-ordered matrices

Steven T. Gabriel, David S. Wise

June 2004 **WMPI '04:** Proceedings of the 3rd workshop on Memory performance issues: in conjunction with the 31st international symposium on computer architecture

Publisher: ACM

Full text available: Pdf (457.69 KB) Additional Information: full citation, abstract, references, cited by, index terms

Bibliometrics: Downloads (6 Weeks): 7, Downloads (12 Months): 18, Downloads (Overall): 131, Citation Count: 3

The Opie Project aims to develop a compiler to transform C codes written for row-major matrix representation into equivalent codes for Morton-order matrix representation, and to apply its techniques to other languages. Accepting a possible reduction ...

Keywords: cache, paging, quadtrees, scientific computing

16 Reducing dynamic compilation overhead by overlapping compilation and execution

P. Unnikrishnan, M. Kandemir, F. Li

January 2006 **ASP-DAC '06:** Proceedings of the 2006 Asia and South Pacific Design Automation Conference

Publisher: IEEE Press

Full text available: Pdf (240.13 KB) Additional Information: full citation, abstract, references, index terms

Bibliometrics: Downloads (6 Weeks): 8, Downloads (12 Months): 24, Downloads (Overall): 101, Citation Count: 0

An important problem in executing applications in energy-sensitive embedded environments is to tune their behavior based on dynamic variations in energy constraints. One option for achieving this is dynamic compilation ---compiling code fragments on ...

17 Evaluation of processor code efficiency for embedded systems

Morgan Hirosuke Miki, Mamoru Sakamoto, Shingo Miyamoto, Yoshinori Takeuchi, Toyohiko Yoshida, Isao Shirakawa

June 2001 **ICS '01:** Proceedings of the 15th international conference on Supercomputing

Publisher: ACM Request Permissions

Full text available: Pdf (205.62 KB) Additional Information: full citation, abstract, references, index terms

Bibliometrics: Downloads (6 Weeks): 7, Downloads (12 Months): 35, Downloads (Overall): 419, Citation Count: 0

This paper evaluates the code efficiency of the ARM, Java, and x86 instruction sets by compiling the SPEC CPU95/ CPU2000/JVM98 and CaffeineMark benchmarks, in terms of code sizes, basic block sizes, instruction distributions, and average instruction ...

Keywords: code efficiency, processor architecture, profiling

18 Compilation and delayed evaluation in APL

Leo J. Guibas, Douglas K. Wyatt

January 1978 **POPL '78:** Proceedings of the 5th ACM SIGACT-SIGPLAN symposium on Principles of programming languages

Publisher: ACM Request Permissions

Full text available: Pdf (944.71 KB) Additional Information: full citation, abstract, references, cited by

Bibliometrics: Downloads (6 Weeks): 3, Downloads (12 Months): 29, Downloads (Overall): 69, Citation Count: 41

Most existing APL implementations are interpretive in nature, that is, each time an APL statement is encountered it is executed by a body of code that is perfectly general, i.e. capable of evaluating any APL expression, and is in no way tailored ...

19 Finding effective compilation sequences

L. Almagor, Keith D. Cooper, Alexander Grosul, Timothy J. Harvey, Steven W. Reeves, Devika Subramanian, Linda Torczon, Todd Waterman

July 2004 **LCTES '04:** Proceedings of the 2004 ACM SIGPLAN/SIGBED conference on Languages, compilers, and tools for embedded systems

Publisher: ACM Request Permissions

Full text available: Pdf (743.88 KB) Additional Information: full citation, abstract, references, cited by, index terms

Bibliometrics: Downloads (6 Weeks): 11, Downloads (12 Months): 66, Downloads (Overall): 606, Citation Count: 23

Most modern compilers operate by applying a fixed, program-independent sequence of optimizations to all programs. Compiler writers choose a single "compilation sequence", or perhaps a couple of compilation sequences. In choosing a sequence, they may ...

Keywords: adaptive compilers, learning models

Also published in:

July 2004 **SIGPLAN Notices** Volume 39 Issue 7

20 Compilation reuse and hybrid compilation: an experiment

Raghavendra Rao Loka

April 2006 **SIGPLAN Notices** , Volume 41 Issue 4

Publisher: ACM

Full text available: Pdf (816.87 KB) Additional Information: full citation, abstract, references, index terms

Bibliometrics: Downloads (6 Weeks): 8, Downloads (12 Months): 14, Downloads (Overall): 58, Citation Count: 0

Compiling hardware models to machine code poses some unusual problems. While compilers for traditional programming languages are well understood, they tend to take very long to compile the C code generated from hardware models. The code generated from ...

Result page: **1** 2 3 4 5 6 next >>>